Molding CNNs for text: non-linear, non-consecutive convolutions

Tao Lei, Regina Barzilay, and Tommi Jaakkola

Presented by Shih-Ming Wang
NLPLab, Institute of Information Science, Academia Sinica

07-5-2016

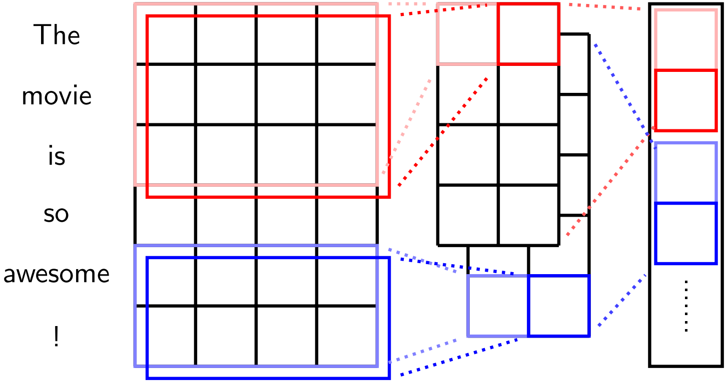
Outline

Introduction

Motivation

- Deep learning & Convolution neural network (CNN) have led to success in many NLP problems
- Convolution operation is a **linear** mapping over **n-gram** vectors
- Target: **non-linear** operation over **non-consecutive** n-grams (e.g., "not that good")

Background



Tensor-based Feature Mapping I

Outer product

- Use outer product operation instead of linear combination
- Consider bi-gram (x_1, x_2) (row vectors) as example:

	Linear	Outer Product	3D case
Raw	$[x_1; x_2]$	$x_1^T \cdot x_2$	$x_1 \bigotimes x_2 \bigotimes x_3$
Dim(raw)	$2 \times d$	$d \times d$	$d \times d \times d$
Dim(Kernel)	$h \times 2 \times d$	$h \times d \times d$	$h \times d \times d \times d$
Output	$h \times 1$	$h \times 1$	$h \times 1$

,where $(x_1 \bigotimes x_2 \bigotimes x_3)_{ijk} = x_{1i} \cdot x_{2j} \cdot x_{3k}$

Tensor-based Feature Mapping II

Parameter Explosion

- Kernel T has $h \times d^n$ parameters for n-gram
- Solution: Decompose T in to sum of $\bar{h}$ rank-1 tensors

	2D	3D
Dim(T)	$h \times d \times d$	$h \times d \times d \times d$
T'	$\sum_{i=1}^{\bar{h}} O_i \bigotimes P_i \bigotimes Q_i$	$\sum_{i=1}^{\bar{h}} O_i \bigotimes P_i \bigotimes Q_i \bigotimes R_i$

,where
$O \in \mathbb{R}^{\bar{h} \times h}$; $P, Q, R \in \mathbb{R}^{h \times d}$;
$O_i \in \mathbb{R}^{h}$; $P_i, Q_i, R_i \in \mathbb{R}^{d}$
For simplity, $\bar{h} = h$.

Tensor-based Feature Mapping III

Feature Map Calculation

	2D	3D
Feature	$x_1 \bigotimes x_2$	$x_1 \bigotimes x_2 \bigotimes x_3$
Kernel	$\sum_{i=1}^{\bar{h}} O_i \bigotimes P_i \bigotimes Q_i$	$\sum_{i=1}^{\bar{h}} O_i \bigotimes P_i \bigotimes Q_i \bigotimes R_i$
Output	$O \cdot (Px_1 \bigodot Qx_2)$	$O \cdot (Px_1 \bigodot Qx_2 \bigodot Rx_3)$

,where $\bigodot$ is element-wise product.

- Px_1 is a linear transformation of x_1
- Higher-order terms (i.e. $x_1 \bigotimes x_2 \bigotimes x_3$) arise from the element-wise products.

Non-consecutive n-gram Features I

Non-consecutive n-gram

- Example: "not nearly as good"
- Intuition: consider all words previous to current word, with decay.

Non-consecutive n-gram Features II

Calculation of non-consecutive n-gram

- Let $z[i,j,k] \in \mathbb{R}^h$ denote the feature corresponding to the 3-gram (x_i, x_j, x_k)
- $z[i,j,k] = O(Px_i \bigodot Qx_j \bigodot Rx_k)$
- Define the **aggregate representation** $z_3[k]$ as a weighted sum of all $z[i,j,k], i < j < k$
- $z_3[k] = \sum_{i<j<k} z[i,j,k] \times \lambda^{(k-j-1)+(j-i-1)}$
- $\lambda \to 0$, the model degrades to traditional 3-gram
- Comment: somehow extends effective window size.

Non-consecutive n-gram Features III

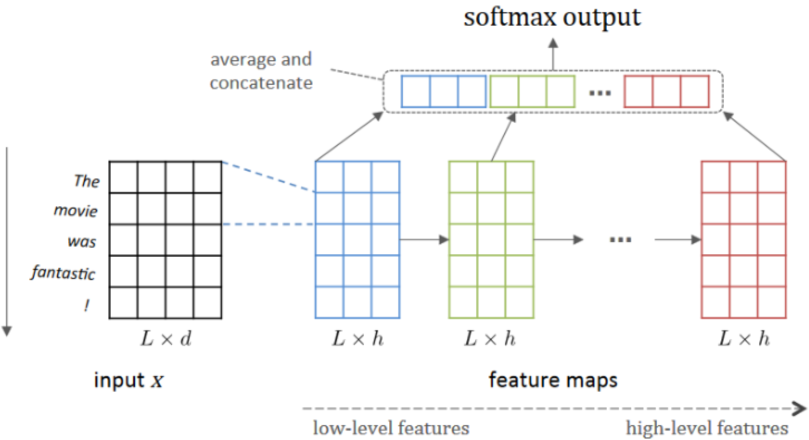
Dynamic Programming

- Calculating all $z_3[k]$ is $O(L^3)$
- In practice, it is calculated as follows:

$$
\begin{aligned}
z_1[k] &= Px_i \\
s_1[k] &= \lambda \cdot s_1[k-1] + f_1[k] \\
z_2[k] &= s_1[k-1] \bigodot Qx_k \\
s_2[k] &= \lambda \cdot s_2[k-1] + f_2[k] \\
z_3[k] &= s_2[k-1] \cdot Rx_k \\
z[k] &= O(z_1[k] + z_2[k] + z_3[k])
\end{aligned}
$$

- Use summation of uni-gram, bi-gram, and tri-gram instead of only tri-gram

Overall Architecture



Experiments I

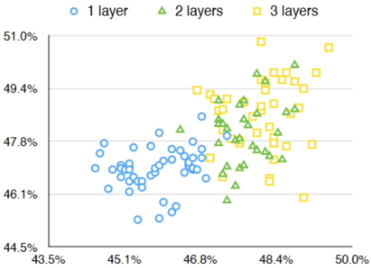
Task

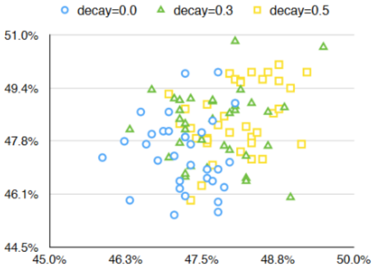
- Sentiment classification
  - Stanford Sentiment Treebank
  - Binary (6920/872/1821) & Fine-grained (5 class) (8544/1101/2210).
- Chinese news categorization
  - Sogou Chinese news corpora
  - 10 news categories (79520/9940/9940)

Experiments II

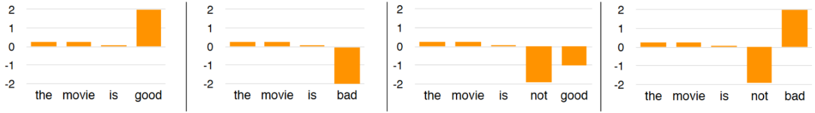
Model	Fine-grained		Binary		Time (in seconds)	
	Dev	Test	Dev	Test	per epoch	per 10k samples
RNN		43.2		82.4	-	-
RNTN		45.7		85.4	1657	1939
DRNN		49.8		86.8	431	504
RLSTM		51.0		88.0	140	164
DCNN		48.5		86.9	-	-
CNN-MC		47.4		88.1	2452	156
CNN	48.8	47.2	85.7	86.2	32	37
PVEC		48.7		87.8	-	-
DAN		48.2		86.8	73	5
SVM	40.1	38.3	78.6	81.3	-	-
NBoW	45.1	44.5	80.7	82.0	1	1
Ours	49.5	50.6	87.0	87.0	28	33
+ phrase labels	53.4	**51.2**	88.9	**88.6**	445	28

Error Analysis I





Error Analysis II



(1) positive prediction (2) negative prediction (3) negative prediction (4) positive prediction

Conclusion

- A feature mapping operator for CNN is proposed
- The method considers non-linear interaction within n-gram
- Non-consecutive n-gram is considered with a weighted sum over previous n-gram
- The method is memory-efficient by factorizing kernel tensor
- The method is time-efficient by adopting dynamic programming
- It achieves state-of-the-art performance